TLM Automation for Multi-core Design

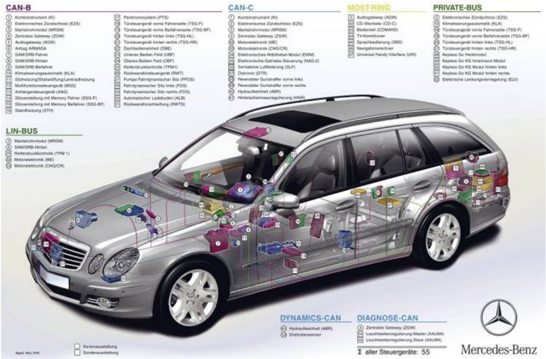
Samar Abdi

Electrical and Computer Engineering
Concordia University, Montreal
`http://www.ece.concordia.ca/~samar`

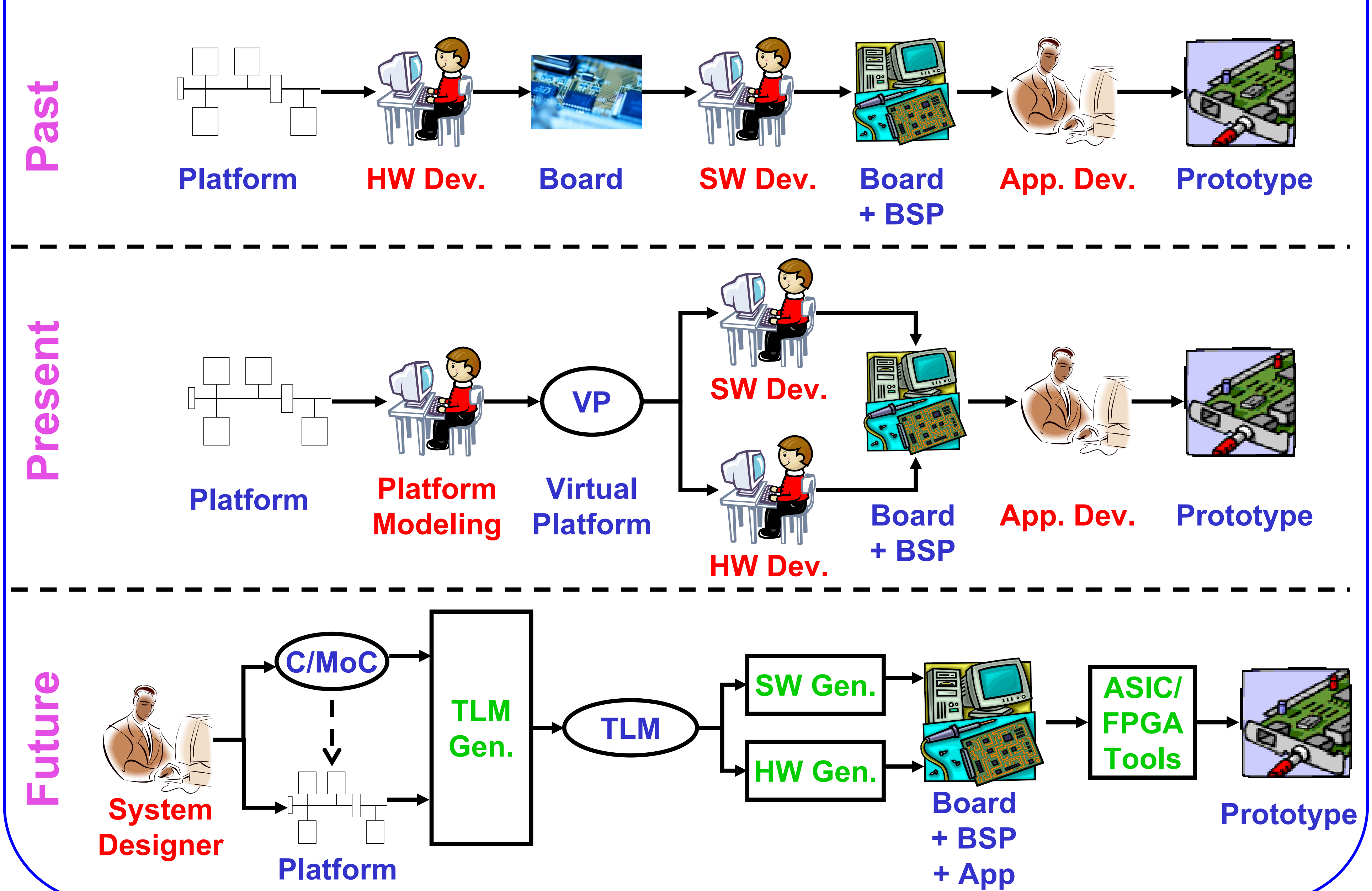
Multi-core Embedded Systems

- **Systems with multiple heterogeneous cores**
  - Application-specific
    - Diverse application areas
  - Tight constraints
    - Timing, throughput, power, size
    - Cost, time-to-market, reliability

- **Ubiquitous and growing**
  - Far bigger market than general-purpose computing (PCs, servers)
    - $46 billion in '04, >$90 billion by 2010, 14% annual growth [BCC]
    - 4 billion devices in '04 [VDC]
    - 99.8% of processors sold [MPR]
- **Rising complexity!**

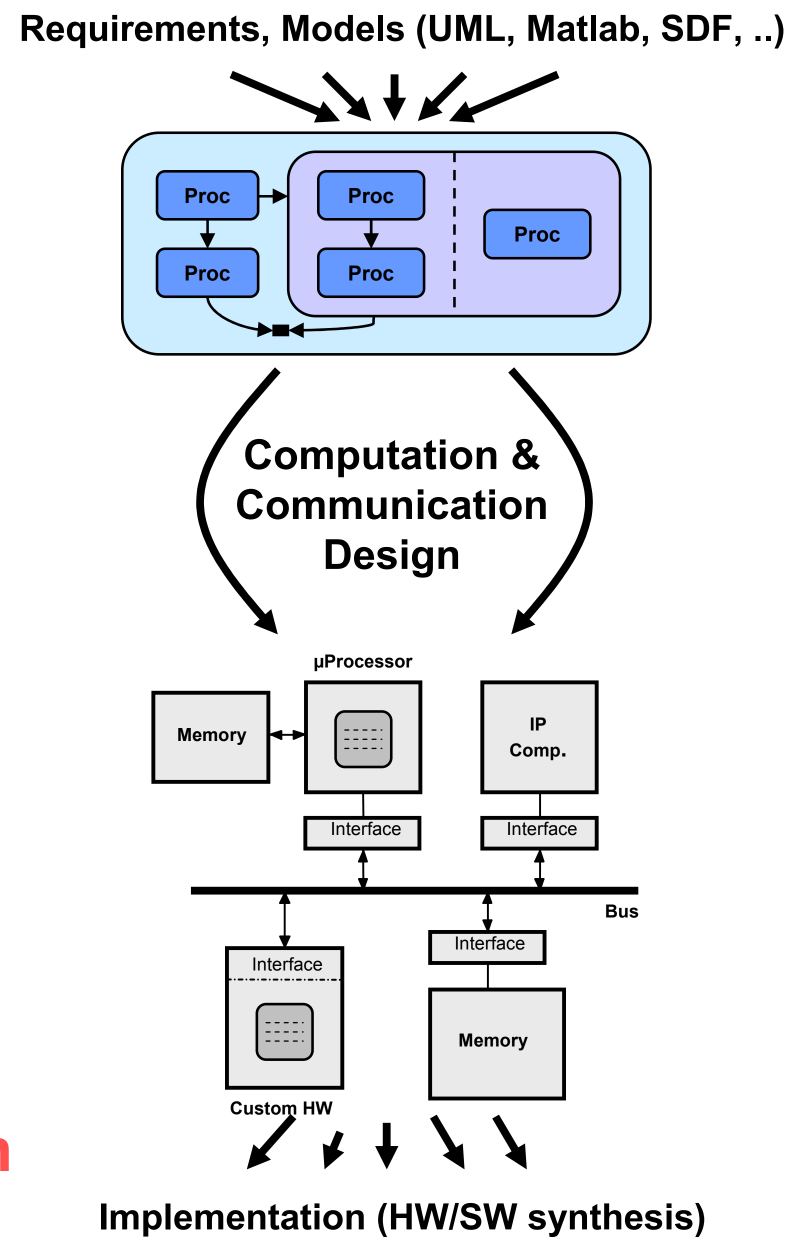


System Design Trends

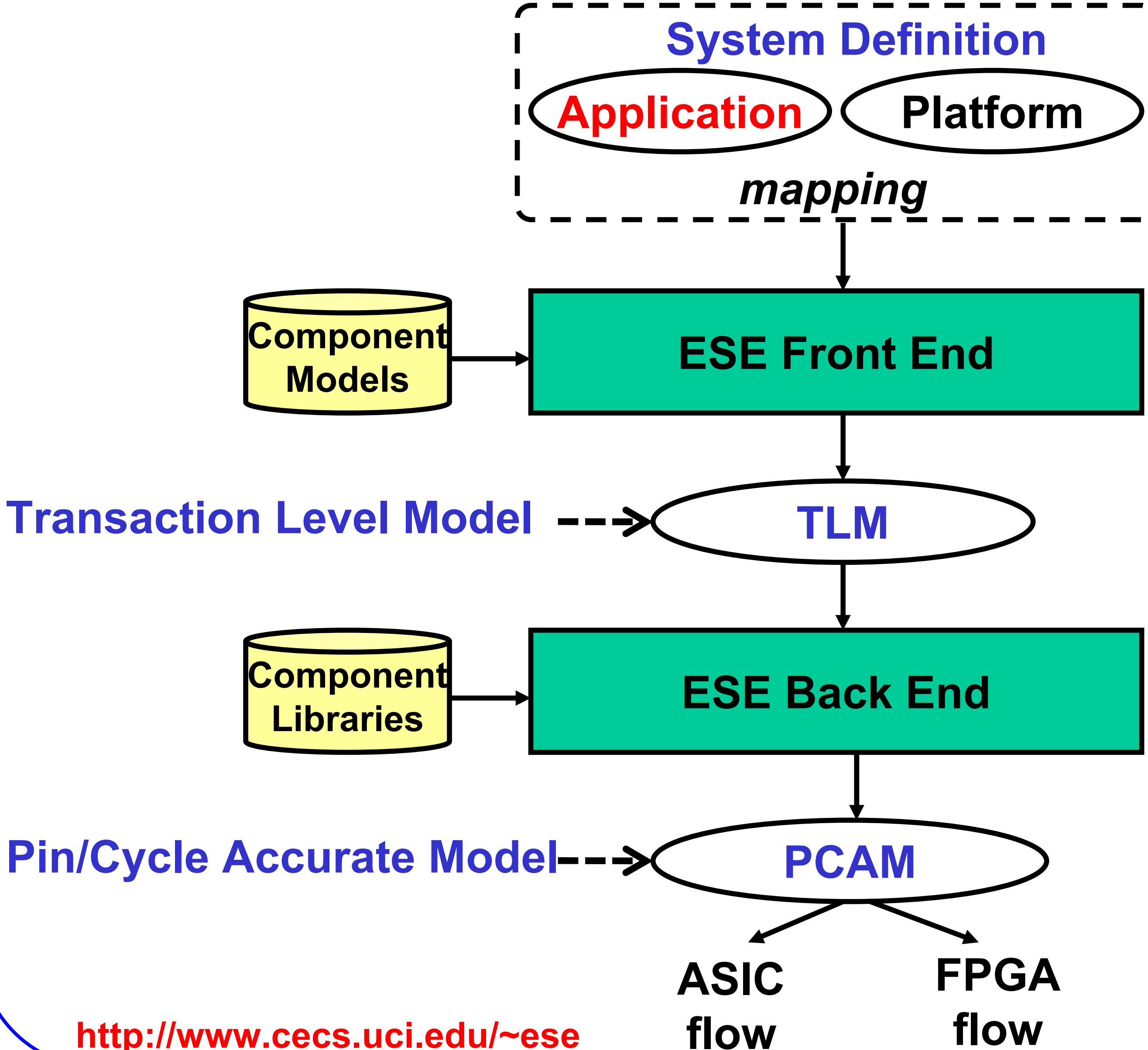


System Level Design Goals

- **From application spec**
  - Functionality, behavior
    - Hierarchical process network
    - Constraints
- **To SW/HW platform**
  - Structure
    - Components and connectivity
    - Mapping from application to platform
- ➢ **New design automation technologies needed**
  - Modeling and analysis
  - Verification
  - Synthesis
- **Enable domain experts to design**

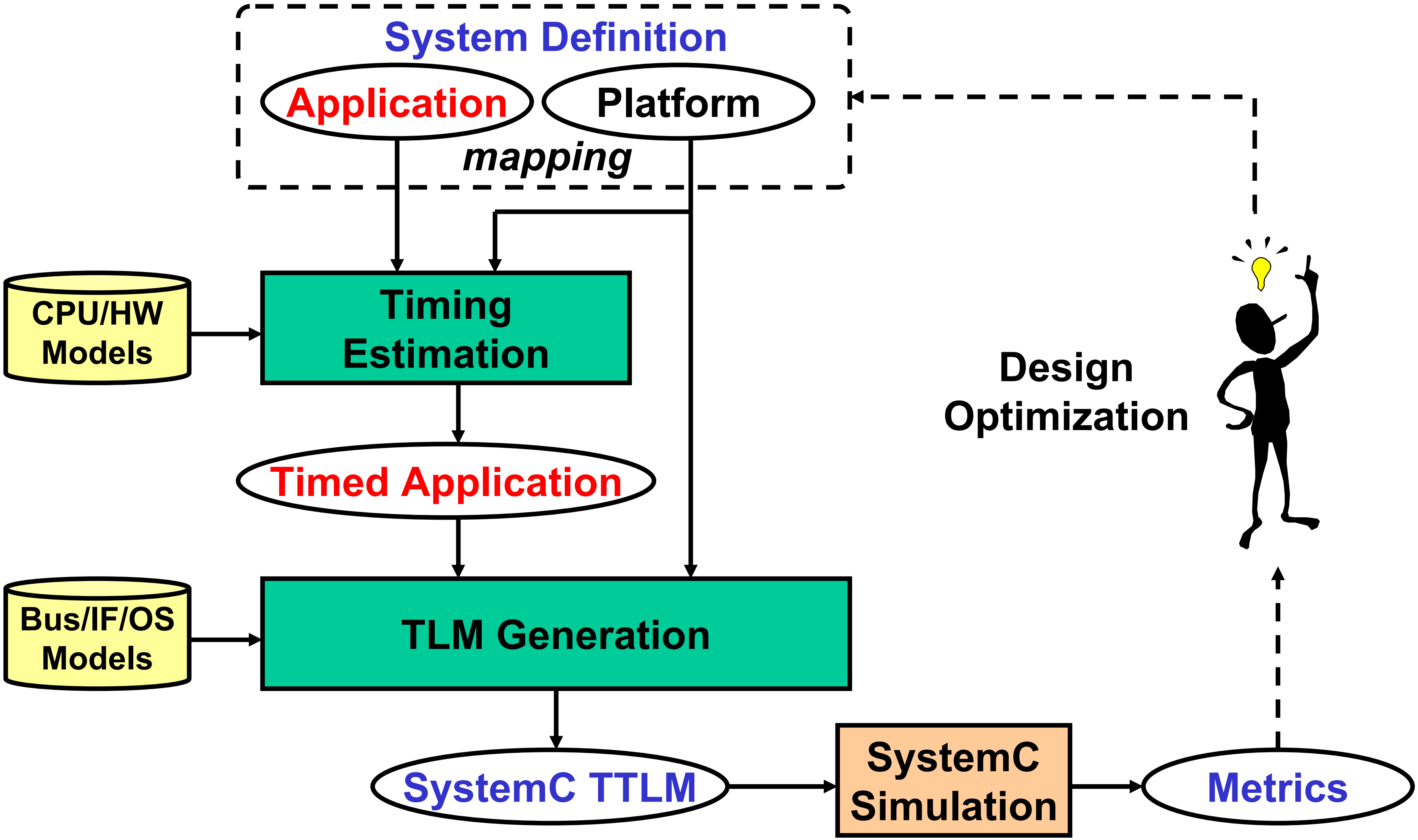


Embedded System Environment (ESE)

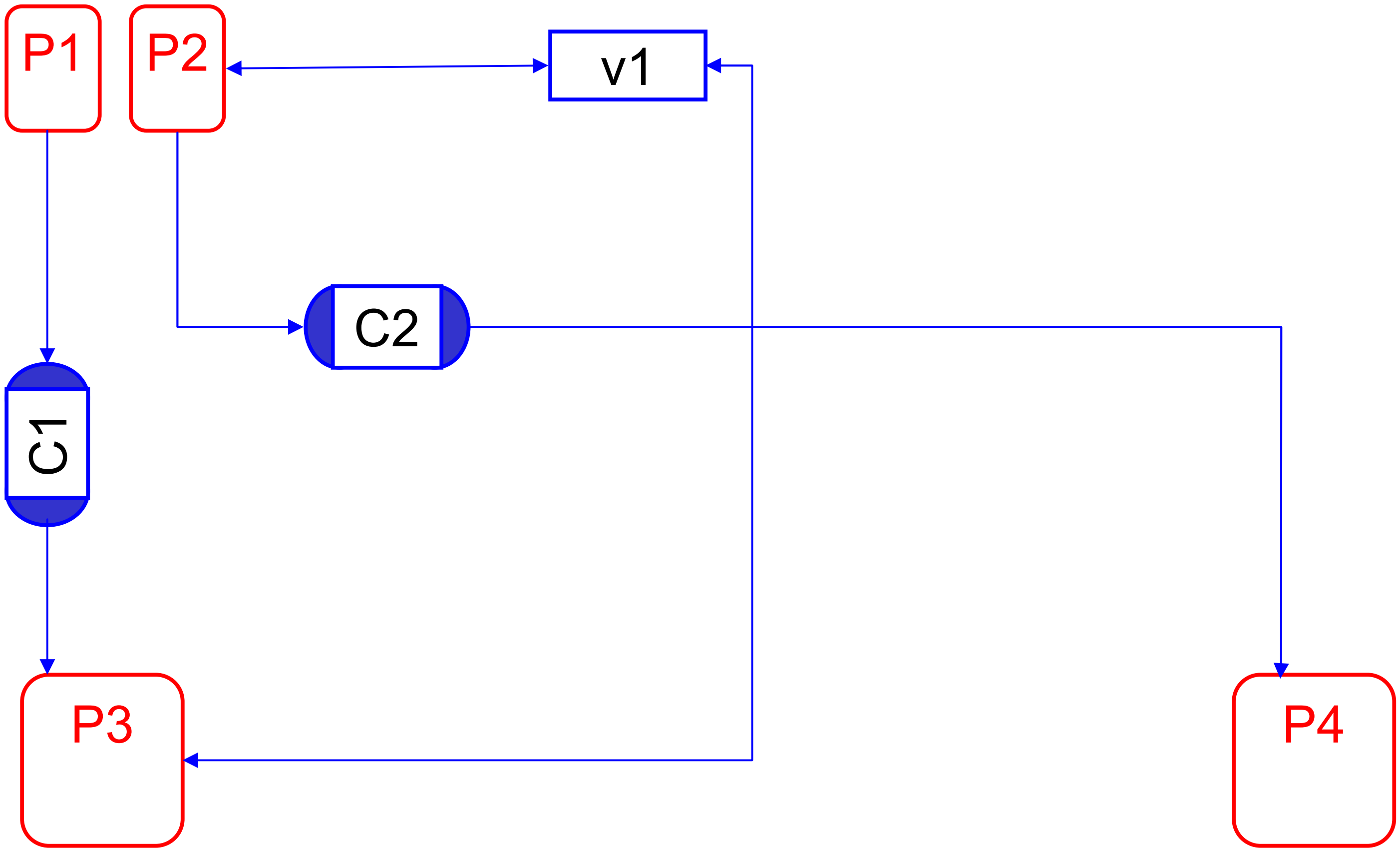


http://www.cecs.uci.edu/~ese

Front End Design Flow

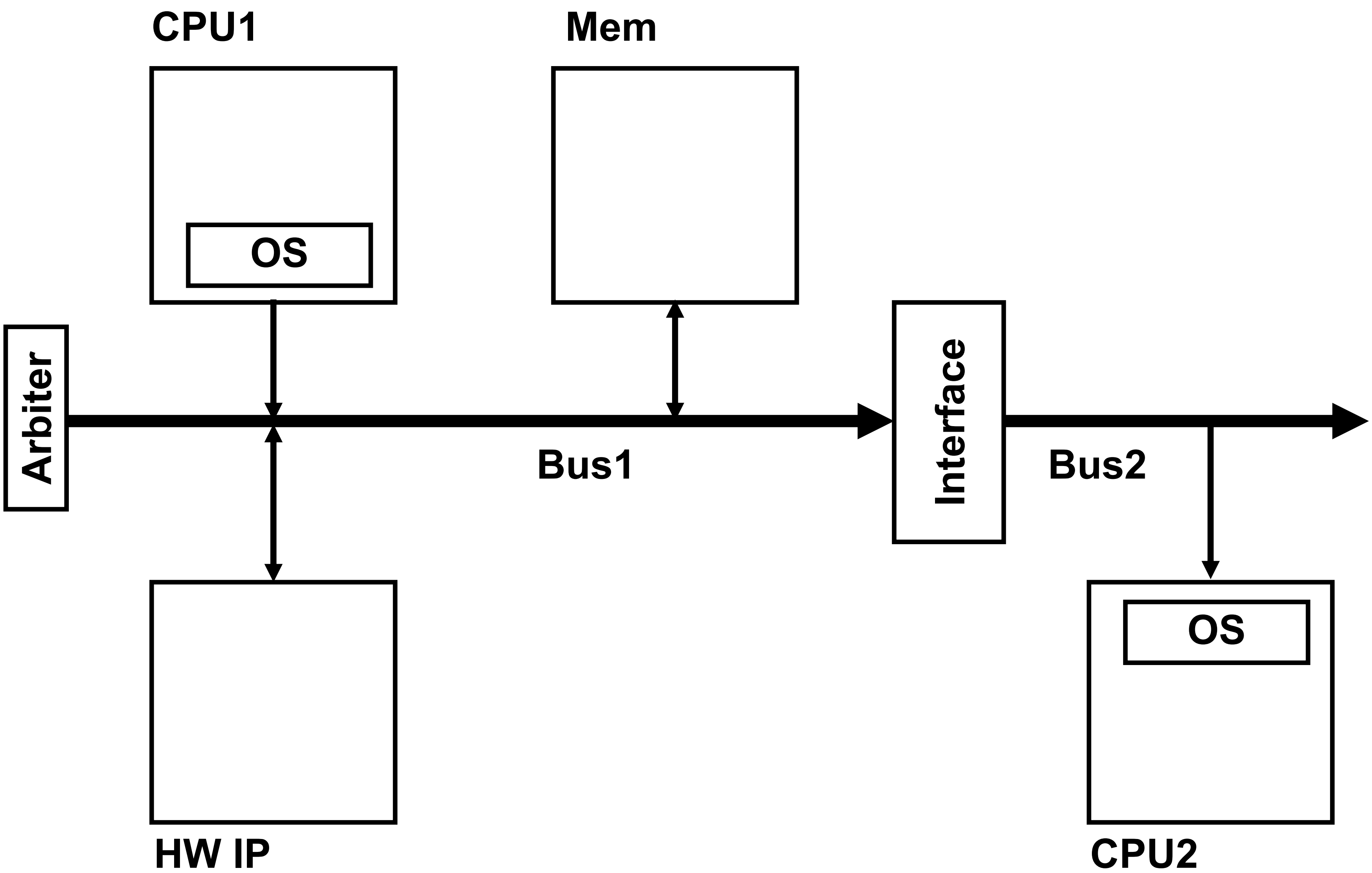


Application Model



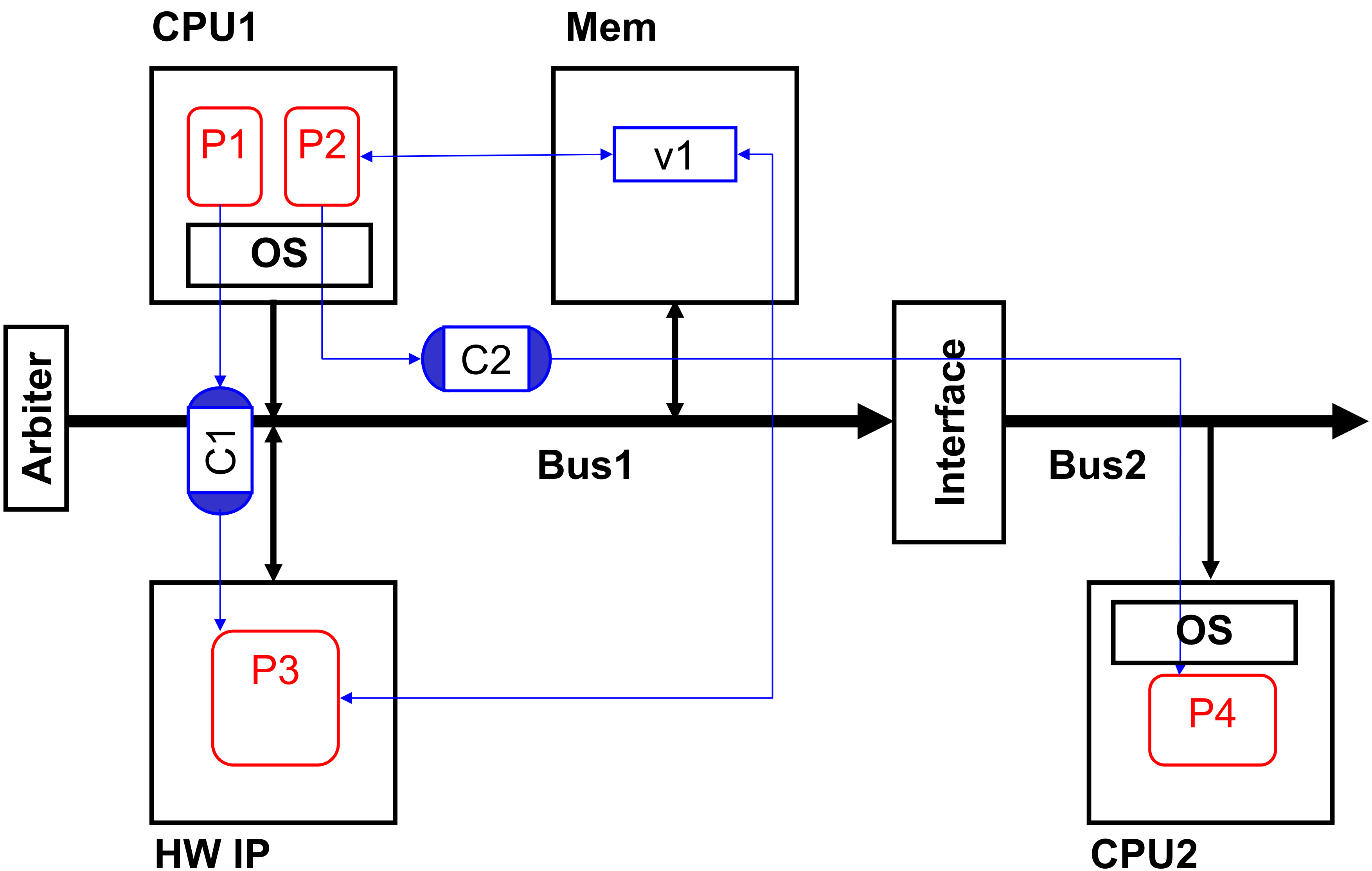
- **Application Spec Model: set of communicating processes**
  - Executes on a simulation kernel (eg: SystemC)
  - Process functionality defined using C/C++
  - Blocking channels and non-blocking variables

Platform Architecture



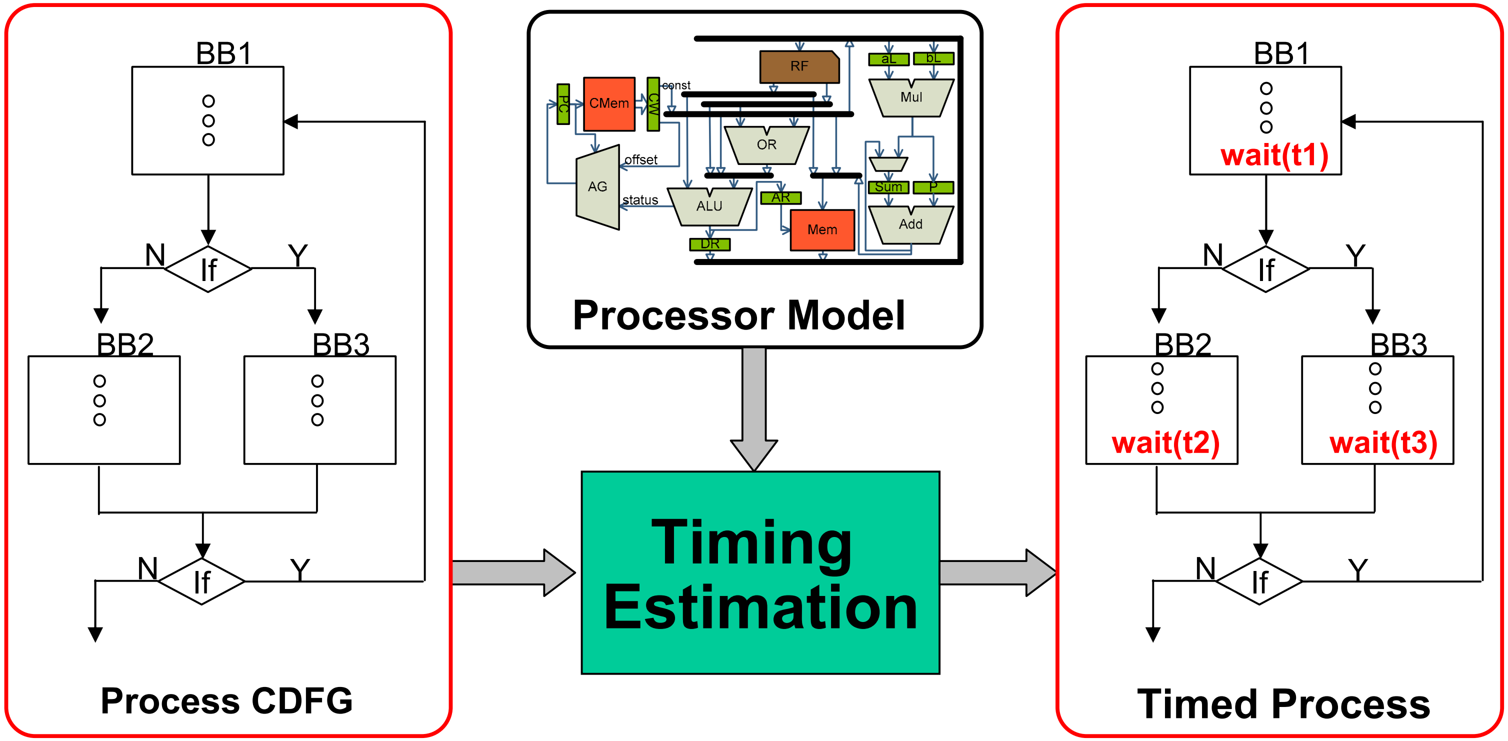
- Netlist of SW processors, HW, buses and interfaces

Input: System Definition



System Definition = ***Platform*** + ***Application***

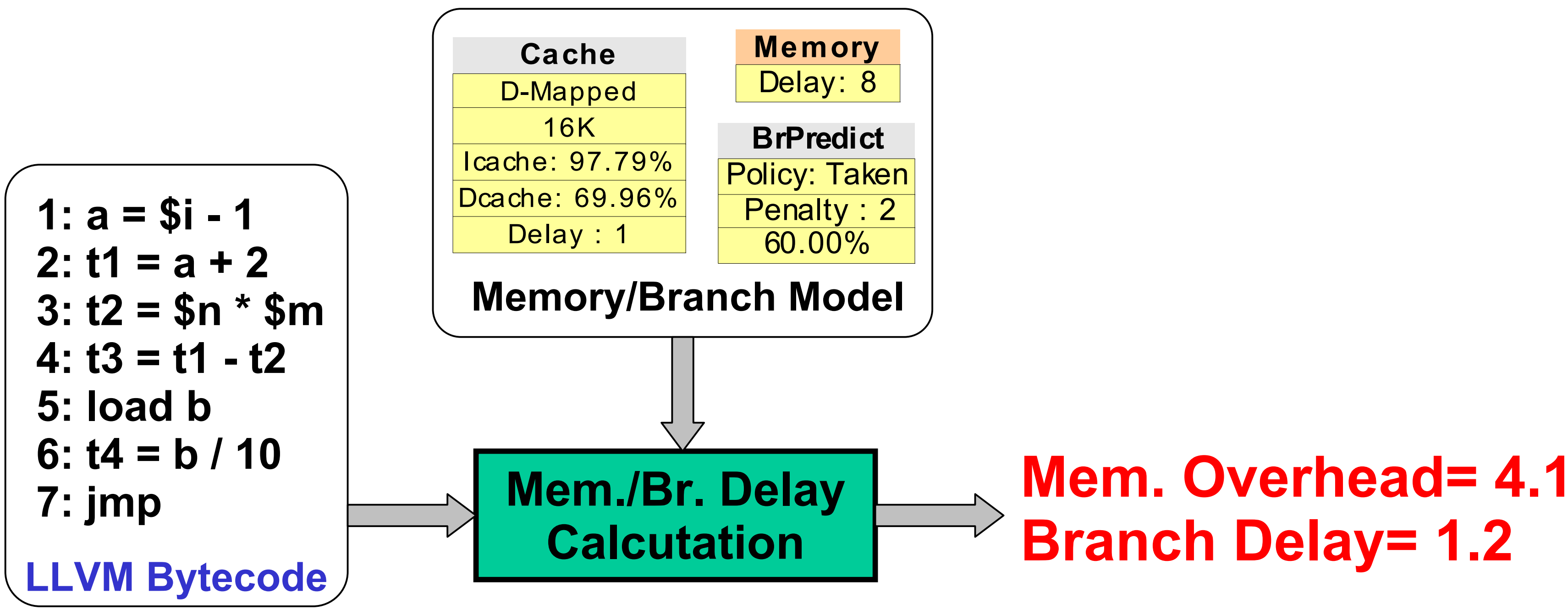
Computation Timing Estimation



- **Stochastic memory delay model**
- **DFG scheduling to compute basic block delay [DATE 08]**
- **RTOS model added for PEs with multiple processes**

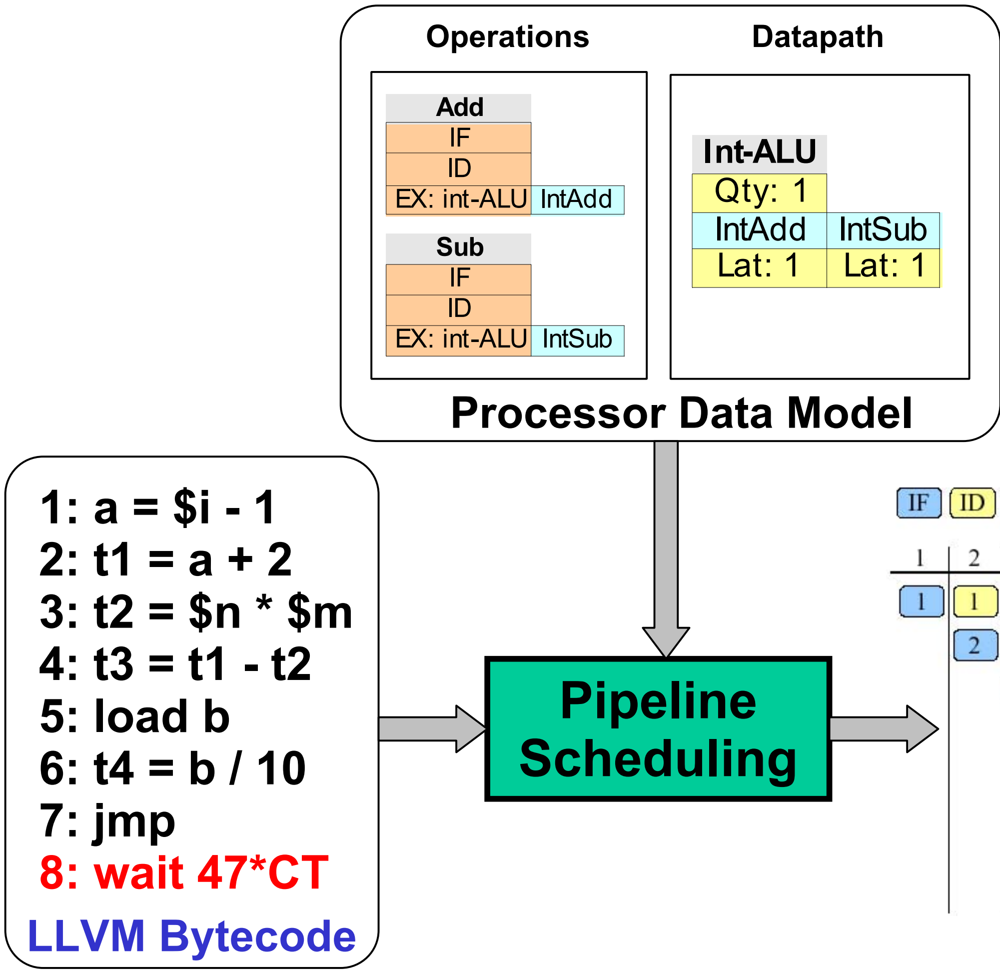
Stochastic Memory Delay Model

- **Assumption**
  - Cache and branch prediction hit rate available in data model
- **Delay Estimation**
  - Operation access overhead = N_{op} * ((1.0 – HR_i) * (CD + L_{mem}))
  - Data access overhead = N_{ld} * ((1.0 – HR_d) * (CD + L_{mem}))
  - Branch prediction miss penalty = MP_{rate} * Penalty



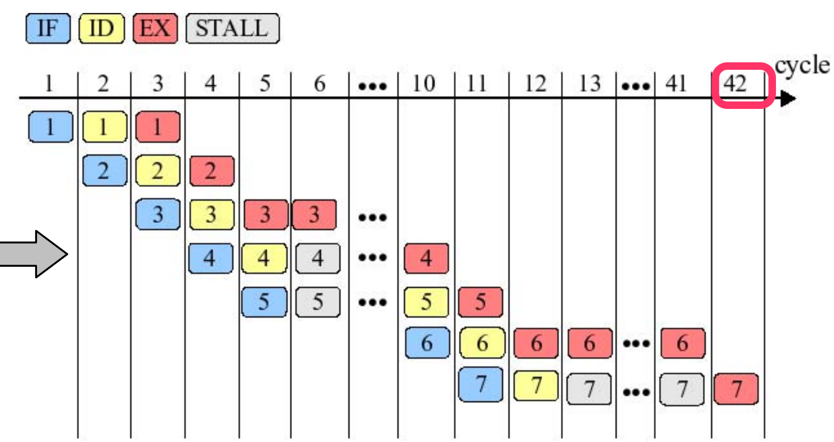
Processor Timing Estimation

- **Assumptions**
  - In-order, single issue processor
  - Optimistic during scheduling (100% cache hit)

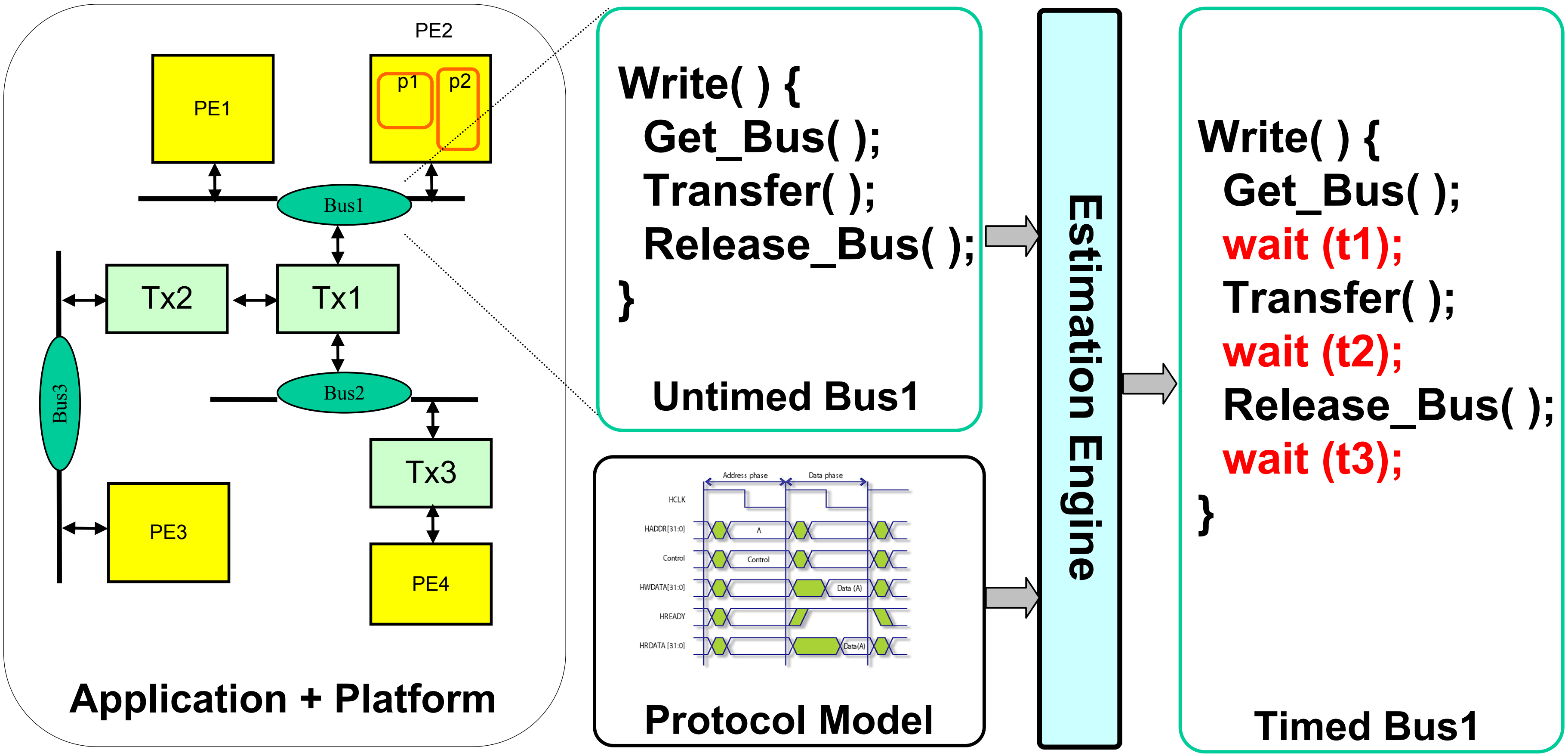


Total BB delay= Op.+Mem.+Br. = 47.3 cycles

Operation delay= 42

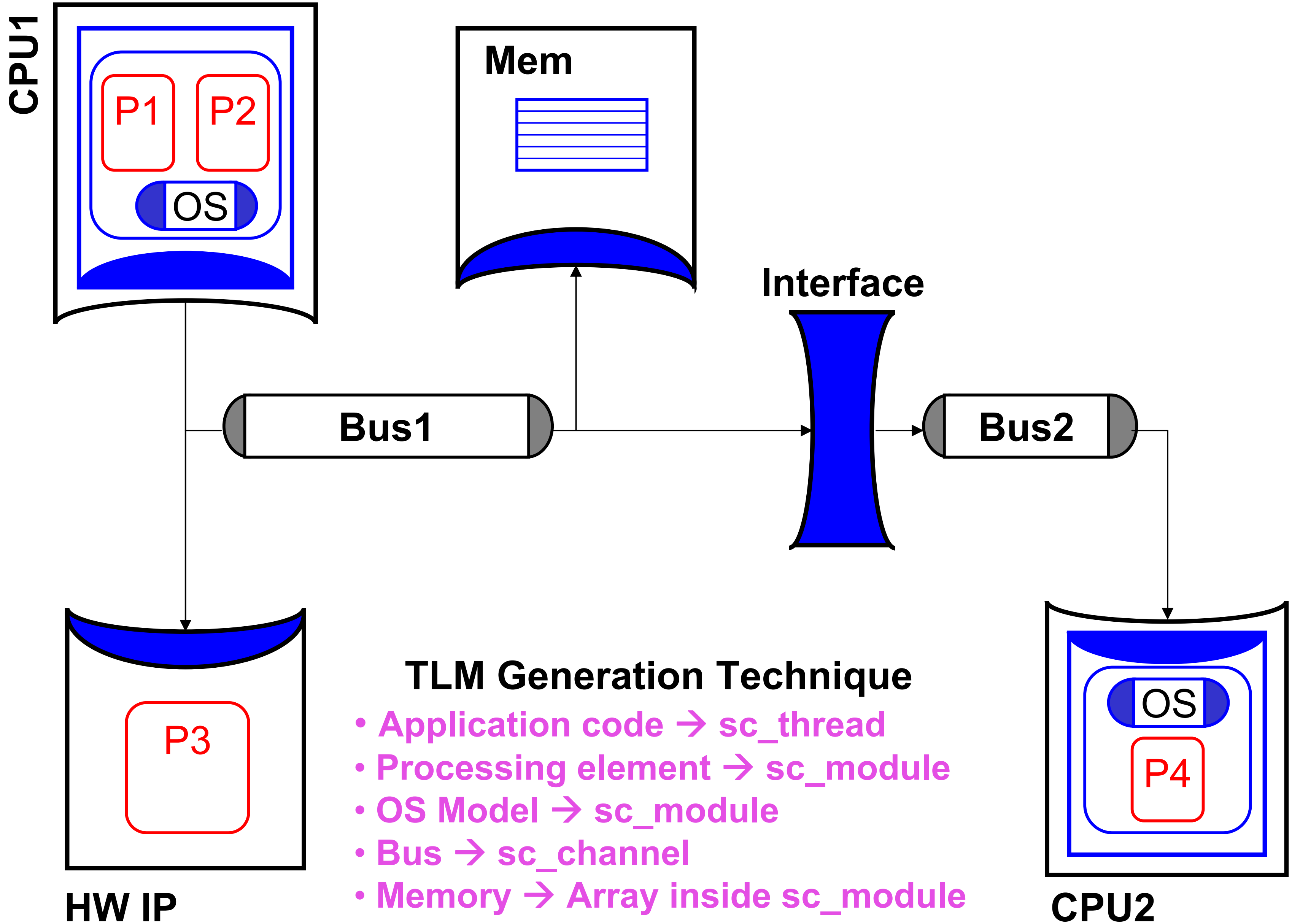


Communication Timing Estimation



- Protocol model used to estimate synchronization, arbitration and transfer [ICCD 2007]
- Timing is annotated in bus channel

Output: SystemC Timed TLM

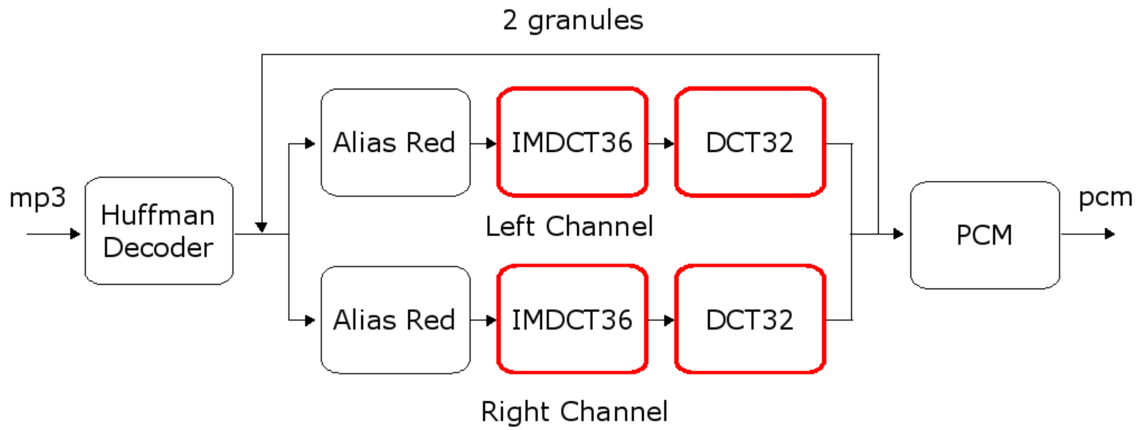


TLM Generation Technique

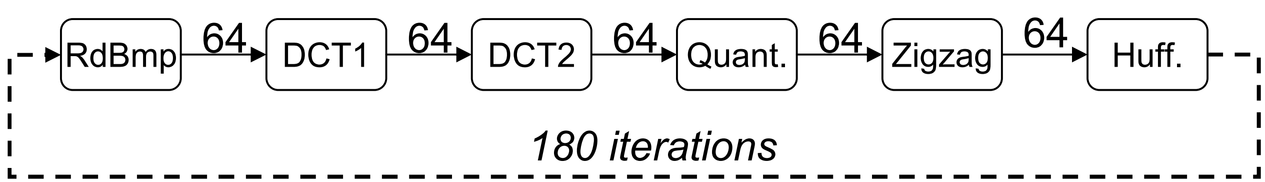
- Application code → sc_thread
- Processing element → sc_module
- OS Model → sc_module
- Bus → sc_channel
- Memory → Array inside sc_module
- Interface → FIFO channel+sc_process

Application Examples: MP3 and JPEG

- **MP3 Decoder Application**
  - 13K lines of C code
  - DCT and IMDCT are compute intensive

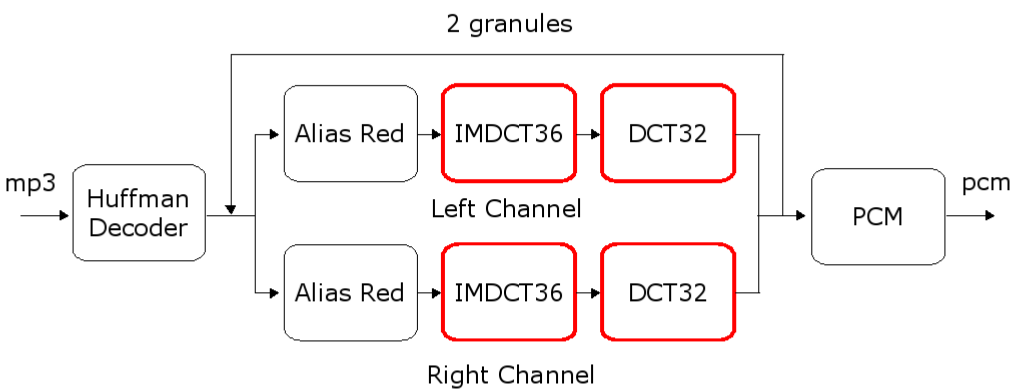


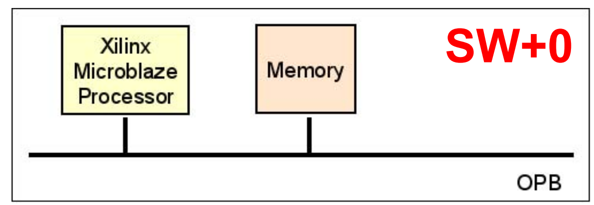
- **JPEG Encoder Application**
  - 2K lines of C code
  - Sequential spec. ideal for pipelining

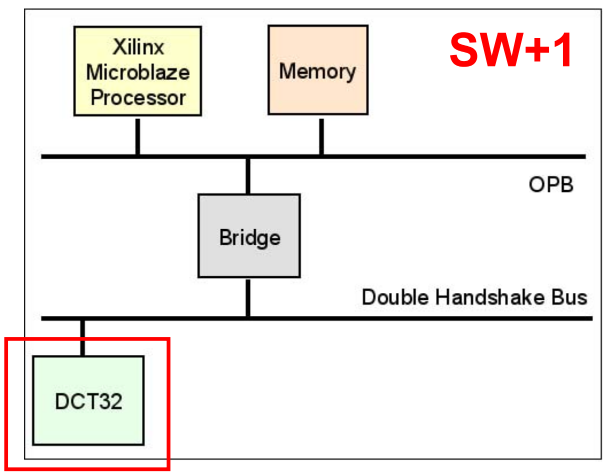


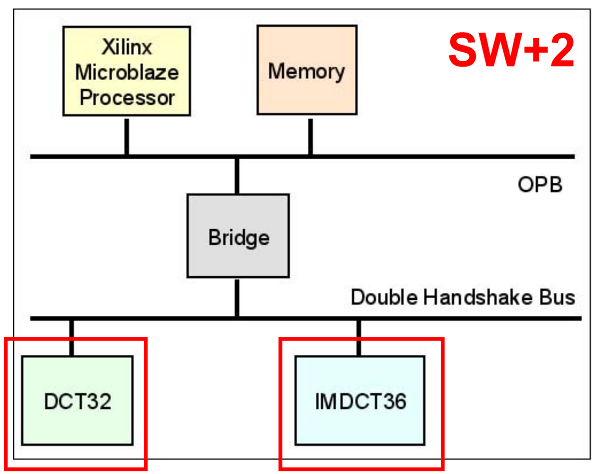
MP3 Platforms

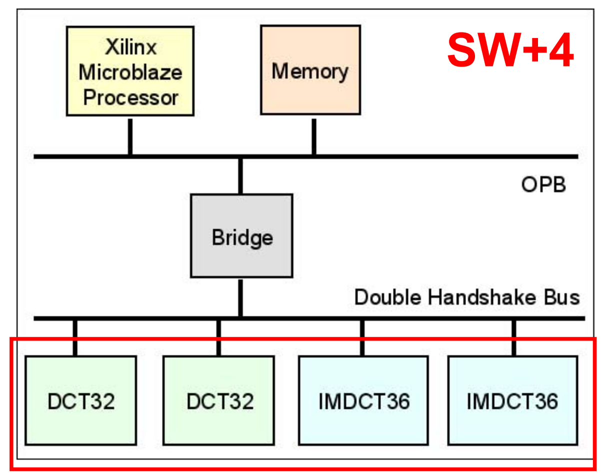
- **MP3 Decoder on Xilinx Multimedia FPGA board**
  - Microblaze soft-core with 0/1/2/4 HW components











MP3 Functional TLM Generation and Simulation

Design	SystemC LoC	Manual Coding	Func. TLM Generation	Func. TLM Simulation
SW+0	2095	2 weeks	0.63 s	0.01 s
SW+1	2894	3 weeks	0.66 s	0.03 s
SW+2	3148	4 weeks	0.66 s	0.12 s
SW+4	3653	4 weeks	0.74 s	0.55 s
Average	**2948**	**~3 weeks**	**~ 0.7 s**	**0.18 s**

- **Functional TLM generation in seconds vs. weeks of manual coding**
  - **Huge productivity gain**
- **Functional TLM simulation in fraction of a second**
  - **Early application development and debugging**

MP3 Timed TLM Generation and Simulation

Design	Timed TLM Generation	Timed TLM Simulation	ISM Sim.	CA Sim.
SW+0	31 s	1 s	3.6 h	16 h
SW+1	50 s	22 s	N/A	18 h
SW+2	47 s	25 s		18 h
SW+4	71 s	36 s		18 h
Average	**~ 1min**	**~ 20 s**	**3.6 h**	**~ 18 h**

- **Timed TLM generated in minutes vs. hours of CA/ISM simulation**
  - **Early SW/HW performance estimation**
- **Timed TLM simulation in < 1 min.**
  - **Extensive design exploration**

MP3 TLM Estimation

Error %= (1 - Estimated cycles/ Board Cycles)*100

ISM Error

Cache size	SW+0	
	Board	ISM Error
0K/0K	27.2M	39.48%
2K/2K	8.9M	18.38%
8K/4K	5.8M	3.55%
16K/16K	4.4M	-16.32%
32K/16K	4.3M	-16.60%
Average	N/A	**18.86%**

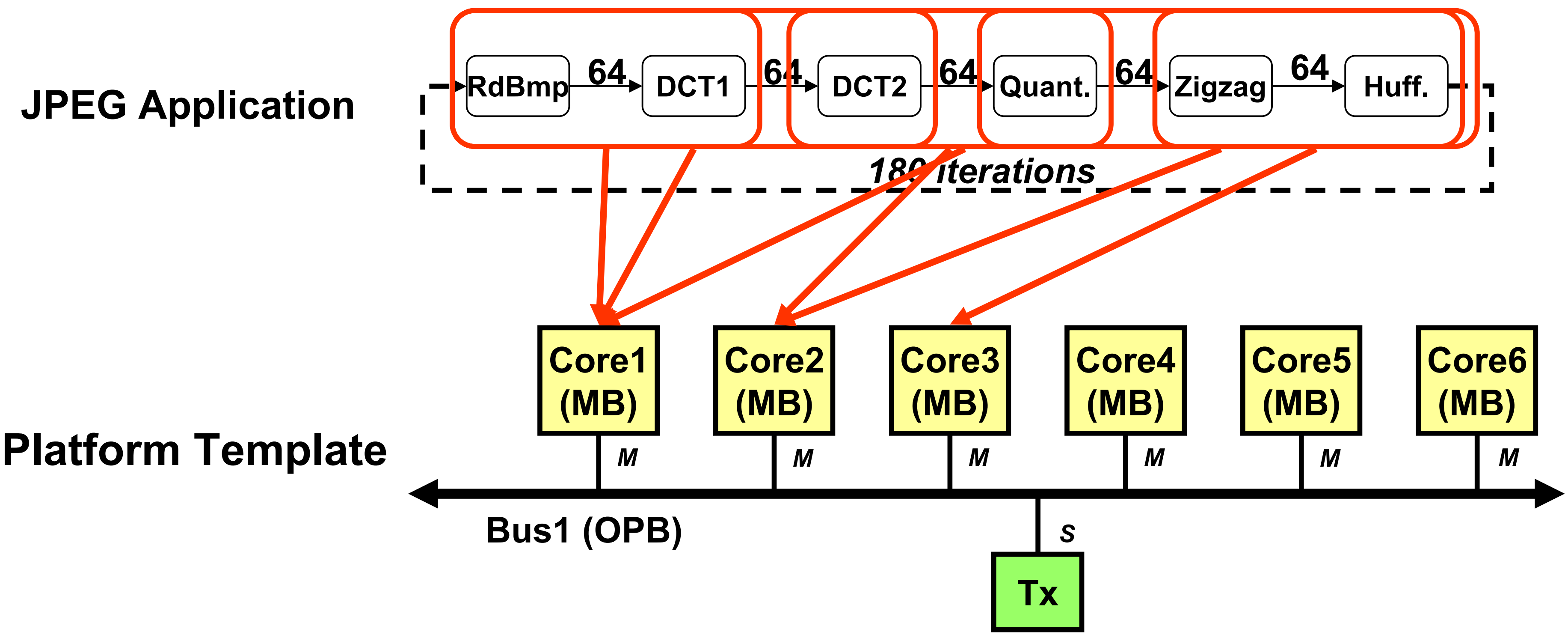
TLM Estimation Error

Cache Size	SW+0	SW+1	SW+2	SW+4
0K/0K	6.27%	9.00%	18.18%	18.61%
2K/2K	6.68%	-7.16%	-15.79%	-9.35%
8K/4K	4.74%	9.13%	-1.66%	-0.18%
16K/16K	-13.83%	4.66%	2.63%	3.65%
32K/16K	-13.89%	-8.29%	1.57%	2.29%
Average	**9.08%**	**7.65%**	**7.97%**	**6.82%**

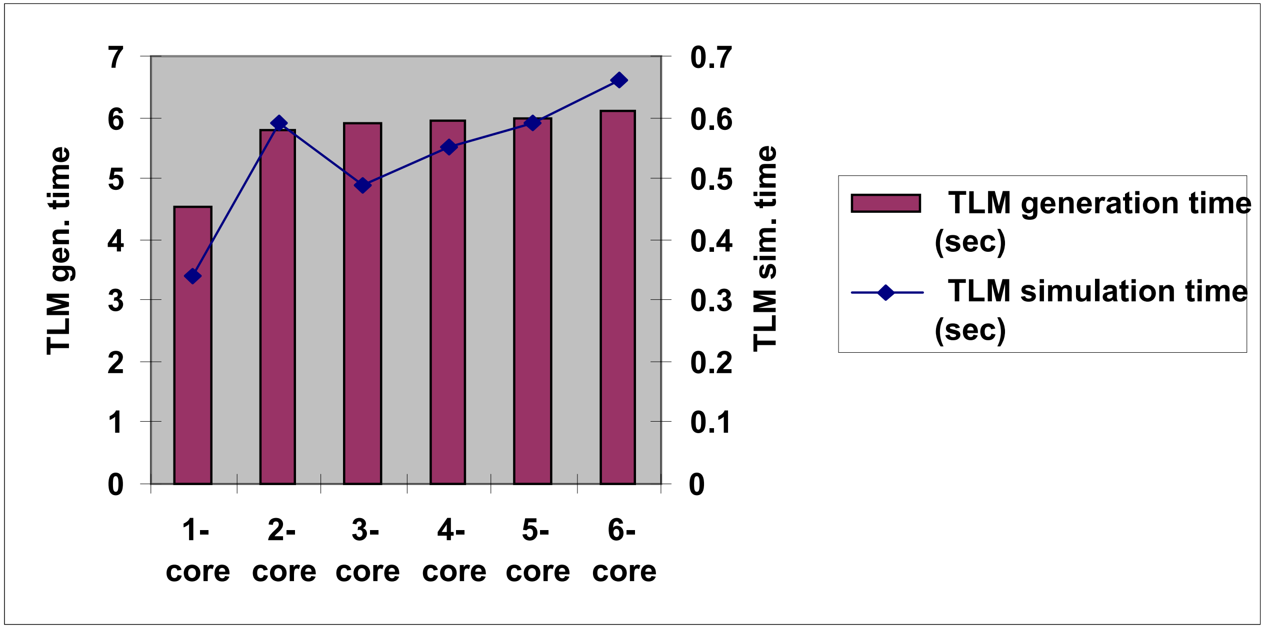
- TLM estimation applicable to all designs
  - ISM only available for SW
- TLM estimation error < ½ of ISM error
  - Reliable design exploration with timed TLMs

JPEG Platforms

- **Multicore platform on Xilinx Virtex-4**
  - 1-6 Microblaze soft-cores with BRAM memory
  - Shared OPB bus with Transducer
  - Application transformed from sequential to pipelined

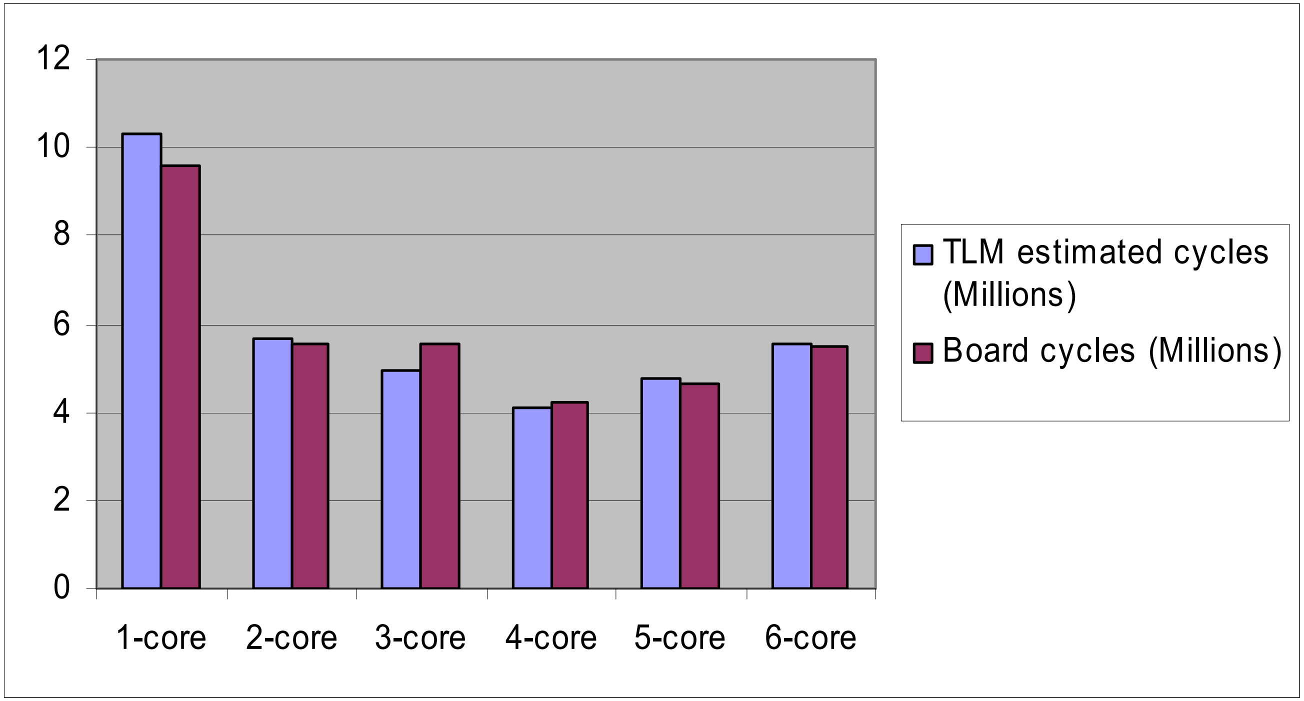


JPEG Timed TLM Generation and Simulation



- **Native TLM simulation is FAST**
  - **<1 sec for all designs**
  - **>3 min. for Xilinx Virtual Platform (XVP) simulation for 1 core**
- **Automatic TLM generation supports EARLY optimization**
  - **SystemC code generation + compilation in <10 sec.**
  - **~ days for RTL coding**

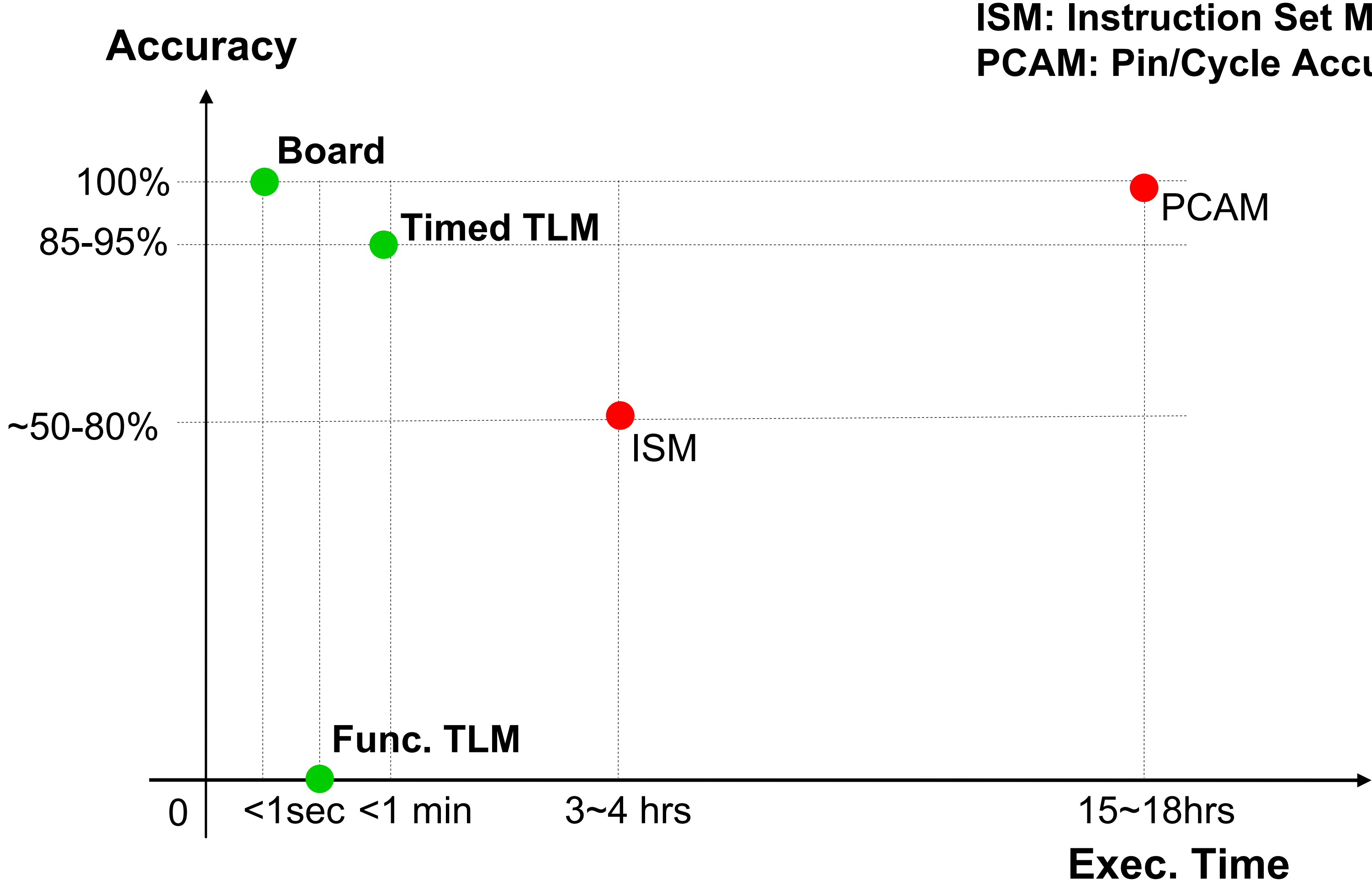
JPEG Timing Estimation



- **Automatically Generated TLM is accurate**
  - **Average error 4.5% compared to board measurement**
  - **Worst case error <11%**
  - **Compare to >50% error using XVP**

Results Summary: TLM vs. Traditional Models

TLM: Transaction Level Model
ISM: Instruction Set Model
PCAM: Pin/Cycle Accurate Model



Time and accuracy trade off among different models

Conclusions

- **Embedded multi-core design flow**
  - Well-defined model semantics
    - Automatic model generation
    - Metric estimation
    - SW/HW synthesis

- **Early and reliable design space exploration**
  - Model generation, compilation and simulation in mins.
  - Accurate modeling of computation and communication

- **Significant productivity gains**
  - Savings in model development and verification costs
  - Huge savings in model debugging costs
  - From system spec to virtual prototype in seconds!

Thank You!

References

- **Automatic TLM generation and estimation**
  - Y. Hwang, S. Abdi, D. Gajski. "*Cycle-approximate retargettable performance estimation at the transaction level,*" Design Automation and Test in Europe, DATE 2008.
  - L. Yu, S. Abdi. "*Automatic TLM Generation for custom communication platforms*," International Conference on Computer Design, ICCD 2007.
- **Software synthesis**
  - D. Gajski, I. Viskic, S. Abdi. "*Model Based Synthesis of Embedded Communication Software*," to appear in Software Technologies for Future Embedded and Ubiquitous Systems, SEUS 2008.
  - I. Viskic, S. Abdi, D. Gajski. "*Automatic generation of embedded communication SW for heterogeneous MPSoC platforms*," Languages, Compilers and Tools for Embedded Systems, LCTES 2007.
- **Interface synthesis**
  - H. Cho, S. Abdi, D. Gajski. "*Interface synthesis for heterogeneous multi-core systems from transaction level models*," Languages, Compilers and Tools for Embedded Systems, LCTES 2007.

